UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/699,144	10/31/2003	Dhruva Ranjan Chakrabarti	200313003-1	3438

22879 7590 12/02/2008
HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
WU, JUNCHUN	

ART UNIT	PAPER NUMBER
2191	

NOTIFICATION DATE	DELIVERY MODE
12/02/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
mkraft@hp.com
ipa.mail@hp.com

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 10/699,144
Filing Date: October 31, 2003
Appellant(s): CHAKRABARTI ET AL.

James K. Okamoto Reg. No. 40,110
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 9/8/2008 appealing from the Office action mailed 6/20/2008 & 1/11/2008.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6195793	Schmidt	2-2001

Ayers et al. "Aggressive Inlining" ACM, 1997

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1, 3-8, and 10- 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over by Ayers et al. ("Aggressive Inlining", 1997, ACM, hereafter "Ayers") and in view of Schmidt (US Patent No. 6,195,793 B1).

3. Per claim 1

Ayers discloses

- A method of compiling a computer program, the method composing:
- receiving a plurality of modules of source code (Fig.1).
- generating intermediate representations corresponding to the modules (Sec.2.1 1st Para. Lines 1-6).
- extracting a set of data from the intermediate representations to create an inliner summary for each module (sec.2.2 1st Para. Lines 1-4 & 3rd Para. Lines 1-5).
- after a call site is determined to be inlined: updating a call graph of the routines and call sites, and updating the inliner summaries throughout the call graph (Sec.2.3 The Last Para.).

But Ayers dose not discloses

- using the inliner summaries and a globally-sorted working-list based order in an inline analysis phase, without using the intermediate representations in the inline analysis phase, to determine which call sites in the modules are to be inlined by substituting code from a called module, wherein said globally-sorted working-list based order is dynamically updated during the inline analysis phase.
- Determining the call sites to be inlined involves proceeding only once through the call sites in said dynamically-updated globally-sorted working-list based order.

However, Schmidt implicitly discloses

- using the inliner summaries and a globally-sorted working-list based order in an inline analysis phase, without using the intermediate representations in the inline analysis phase, to determine which call sites in the modules are to be inlined by substituting code from a called module (col.3 lines 42~44 & lines 53-62).
- Determining the call sites to be inlined involves proceeding only once through the call sites (col.3 lines 42-53 *"A more global view of inlining effects is provided in order to select good inlining candidates while accurately spending the code bloat budget. First the best call sites at which to inline are estimated, based upon the execution frequencies of the call sites and the sizes of the called procedures. Then the call graph is processed starting from the leaves and working up. Each time an arc that was selected for inlining is encountered, the original bloat estimate is compared with the current size of the procedure, for example, incorporating sizes of any procedures that were inlined into the*

procedure. If the called procedure has been bloated beyond an acceptable limitation, it may be rejected for inlining.").

- dynamically updated the working list of call sites during the incline analysis phase (col.7 lines 31-38 *"If the call site exceeds the allowable growth at decision block 408, its priority is recalculated based on the bloat now known that will be incurred as indicated at a block 410. A best call site j residing in the AuxQueue is obtained as indicated at a block 412. Then checking to see whether the best call site j residing in the AuxQueue obtained at block 412 has a priority at least as great as the one being considering is performed as indicated at a decision block 414. ").*
- Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Ayer's teachings by adding using the inliner summaries and a globally-sorted working-list based order in an inline analysis phase, without using the intermediate representations in the inline analysis phase, to determine which call sites in the modules are to be inlined by substituting code from a called module, wherein said globally-sorted working-list based order is dynamically updated during the inline analysis phase as taught by Schmidt in order to select good inlining candidates by an adaptive approach in accordance with features of preferred embodiment (col.3 lines 40-42).

4. Per claims 4 and 11

the rejection of claim 3 and 10 are incorporated respectively and Ayers discloses

- updating the inliner summaries comprises determining nodes and edges of the call graph that are affected by the inlining of the call site and updating those inliner summaries corresponding to the affected nodes and edges (Sec.2.3 The Last Para. Lines 5-8).

5. Per claims 5 and 12

the rejection of claim 4 and 11 are incorporated respectively and Ayers discloses

- the edge summaries include at least a call site execution count and a signature type (Sec.2.3 1st Para.).

6. Per claims 7

the rejection of claim 1 is incorporated and Ayers further discloses

- the inline analysis phase is separate and distinct from an inline transformation phase (Sec. 2.3 1st & 2nd Para. (i.e. analysis phase for determining which call site is clonable); Sec.2.3 6th & 7th Para. (i.e. transformation phase for using the results of analysis phase to create clones and fix call sites).

7. Per claim 8

Ayers discloses

- An apparatus for compiling a computer program, the apparatus comprising:
  A processor configured to execute computer-readable code; a memory system configured to store data; computer-readable code for a front-end portion of compiler program, the front-end portion of the compiler program being configured to receive a plurality of

modules of source code (Fig.1), generate intermediate representations corresponding to the modules (Sec.2.1 1st Para. Lines 1-6), and extract a set of data from the intermediate representations to generate inliner summaries for the modules (sec.2.2 1st Para. Lines 1-4 & 3rd Para. Lines 1-5).

- after a call site is determined to be inlined: updating a call graph of the routines and call sites, and updating the inliner summaries throughout the call graph (Sec.2.3 The Last Para.).

Schmidt discloses

- Computer-readable code for a cross-module optimizer of the compiler program, the cross-module optimizer being configured to use the inliner summaries and a dynamically-updated globally-sorted working-list based order to analyze the call sites in an inline analysis phase, without using the intermediate representation, so as to determine which call sites in the modules are to be inlined by substituting code from a called module (col.3 lines 42~44 & lines 53-62).
- Determining the call sites to be inlined involves proceeding only once through the call sites (col.3 lines 42-53 *"A more global view of inlining effects is provided in order to select good inlining candidates while accurately spending the code bloat budget. First the best call sites at which to inline are estimated, based upon the execution frequencies of the call sites and the sizes of the called procedures. Then the call graph is processed starting from the leaves and working up. Each time an arc that was selected for inlining is encountered, the original bloat estimate is compared with the current size of the procedure, for example, incorporating sizes of any procedures that were inlined into the*

procedure. If the called procedure has been bloated beyond an acceptable limitation, it may be rejected for inlining.").

- dynamically updated the working list of call sites during the incline analysis phase (col.7 lines 31-38 *"If the call site exceeds the allowable growth at decision block 408, its priority is recalculated based on the bloat now known that will be incurred as indicated at a block 410. A best call site j residing in the AuxQueue is obtained as indicated at a block 412. Then checking to see whether the best call site j residing in the AuxQueue obtained at block 412 has a priority at least as great as the one being considering is performed as indicated at a decision block 414.* ").

8. Per claim 14

the rejection of claim 8 is incorporated and Ayers further discloses

- the inline analysis phase is separate and distinct from an inline transformation phase (Sec. 2.3 1st & 2nd Para. (i.e. analysis phase for determining which call site is clonable); Sec.2.3 6th & 7th Para. (i.e. transformation phase for using the results of analysis phase to create clones and fix call sites).

9. Per claim 15

Ayers discloses

- after a call site is determined to be inlined: updating a call graph of the routines and call sites, and updating the inliner summaries throughout the call graph (Sec.2.3 The Last Para.).

Schmidt discloses

- a computer program product comprising a computer-usable medium having computer-readable code embodied therein, the computer program product being compiled from a plurality of modules of source code using inliner summaries and a dynamically updated globally-sorted working-list based order in an inline analysis phase, without using intermediate representations, to determine which call sites in the modules are to be inlined by substituting code from a called module (col.3 lines 42~44 & lines 53-62).
- Determining the call sites to be inlined involves proceeding only once through the call sites (col.3 lines 42-53 *"A more global view of inlining effects is provided in order to select good inlining candidates while accurately spending the code bloat budget. First the best call sites at which to inline are estimated, based upon the execution frequencies of the call sites and the sizes of the called procedures. Then the call graph is processed starting from the leaves and working up. Each time an arc that was selected for inlining is encountered, the original bloat estimate is compared with the current size of the procedure, for example, incorporating sizes of any procedures that were inlined into the procedure. If the called procedure has been bloated beyond an acceptable limitation, it may be rejected for inlining."*).
- dynamically updated the working list of call sites during the incline analysis phase (col.7 lines 31-38 *"If the call site exceeds the allowable growth at decision block 408, its priority is recalculated based on the bloat now known that will be incurred as indicated at a block 410. A best call site j residing in the AuxQueue is obtained as indicated at a block 412. Then checking to see whether the best call site j residing in the AuxQueue*

obtained at block 412 has a priority at least as great as the one being considering is performed as indicated at a decision block 414. ").

10. Per claim 16

Schmidt discloses

A method of compiling a computer program with a plurality of modules of source code and corresponding intermediate representations (see Fig.1), the method comprising:

- using the inliner summaries in a one-pass inline analysis phase, without using the intermediate representations, to determine which call sites to inline, in what order to inline them, and preserving a same order during the transformation phase (col.2 lines 30~36 "*A first approximation of initial call sites of the identified possible call sites are identified for inlining. Procedures in the call multigraph are processed in a determined order where a first procedure is only processed after all second procedures called by the first procedure are processed.*").
- formulating a measure of goodness for each call site from the inliner summary (col.3 lines 40-44).
- using a technique to dynamically update information in the summary for potentially all call-graph nodes and edges every time a call site is accepted for inlining (col.3 lines 45-50).
- using a globally sorted work-list and an associated table to maintain and manipulate the call sites and dynamically updating the work-list every time a call site is accepted for inlining (col.3 lines 52-62).

Ayers discloses

- extracting a set of data from the intermediate representations of the modules to create an inliner summary for each module (sec.2.2 1st Para. Lines 1-4 & 3rd Para. Lines 1-5).

11. Per claim 17

the rejection of claim 16 is incorporated and Schmidt further discloses

- wherein the inliner summaries are comprised of: a code size (col.6 lines 3~4); a call site profile count (col.5 lines 30-33); a critical path length; an execution time (col.3 lines 44-46); a node level; a level criticality; and a total execution count (col.4 lines 30~32).

12. Per claim 18

- the rejection of claim 16 is incorporated and Schmidt further discloses

  wherein an arbitrary inlining order among the call sites in the call graph is selectable in an inline analysis phase, and wherein that same order is followed in an inline transformation phase (col.4 lines 2-6 "*the fundamental concept of the inline candidate selection method is to select an initial set of inline candidates and adaptively modify the initial set of inline candidates as more information becomes available.*").

13. Per claim 19

- the rejection of claim 16 is incorporated and Schmidt further discloses

  wherein a measure of goodness for each call site is computed from the light-weight inliner summary, and a total profit is computed as a product of component profit factors, and a total cost is computed as a product of component cost factors (col.5 lines 21-25 "*The amount of code bloat that appropriately can incur is calculated by adding the total estimated instruction stream sizes for all procedures and multiplying by a tunable percentage factor, bloat factor as indicated at a block 202.*").

14. Per claim 20

- the rejection of claim 16 is incorporated and Schmidt further discloses
  wherein information in the light-weight inliner summary corresponding to call-graph nodes and edges are updated each time a call site is accepted for inlining, and wherein a goodness factor of each call site with updated summary information is re-computed (col.7 lines 31-33).

15. Per claim 21

- the rejection of claim 16 is incorporated and Schmidt further discloses
  wherein a globally-sorted work list and an associated table are maintained and used to continuously order the work list and extract a call site with a highest goodness factor (col.6 lines 8-13)

16. Per claims 3 and 10

the rejection of claim 1 and 8 are incorporated respectively

Ayers does not teach

- after the call graph and inliner summaries are updated, re-calculating profitabilities associated with remaining call sites; and re-ordering the working list using the re-calculated profitabilities.

But Schmidt teaches

- re-calculating profitabilities associated with remaining call sites; and re-ordering the working list using the re-calculated profitabilities (Schmidt, col.7 lines 31-65).
- Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Ayer's teachings by adding re-calculating profitabilities associated with remaining call sites; and re-ordering the working list using the re-calculated profitabilities as taught by Schmidt in order to determine whether the alternate call site from working list (i.e. AuxQueue) should be inlined if the priority best call site in AuxQueue is less than a threshold that is acceptable for the original call site after the re-calculating (Schmidt, col.7 lines 39-51).

17. Per claims 6 and 13

the rejection of claim 4 and 11 are incorporated respectively

Ayers does not teach

- the node summaries include at least a code size, a routine execution count, and a call-graph height.

But Schmidt teaches

- the node summaries include at least a code size, a routine execution count, and a call-graph height (Schmidt, col.4 lines 26-34).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Ayer's teachings by adding the node summaries include at least a code size, a routine execution count, and a call-graph height as taught by Schmidt in order to select good inlining and making accurate estimates of code bloat (Schmidt, col.3 lines 42-50).

(10) Response to Argument

The rejection is elaborated for clarification purpose.

Appellant has argued in the following:

1. In regard to independent claims 1, 8, 15 and 16, the cited references do not teach "dynamically updating the order of call sites in the working list".

Examiner Responses:

According to Schmidt's disclosure, Schmidt teaches using priority-sorted list throughout procedures to performance the inline analysis. As stated in col.6 lines 1-20, in the refinement stage, Schmidt teaches using ReadyQueue that contains all procedures that can be issued next in the compilation order and AuxQueue that contains initially outlined call sites whose final bloat is known. But Schmidt further discloses as stated in col.6 lines 46- 52 and col.7 lines 31- 38 "If the outgoing arc was selected as an inline candidate, checking to see whether the target procedure has grown in size since that decision was made is performed as indicated at a decision block 408, and if so by how much. A tunable parameter, GrowthThreshold utilized at decision block 408 determines how much growth is permissible before an alternate inlining candidate should be sought. If the call site exceeds the allowable growth at decision block 408, its priority is <u>recalculated</u> based on the bloat now known that will be incurred as indicated at a block 410. A best call site j residing in the AuxQueue is obtained as indicated at a block 412. Then checking to see whether the best call site j residing in the AuxQueue obtained at block 412 has a priority at least as great as the one being considering is performed as indicated at a decision block 414". That is, during inline refinement phase, the parameter GrowthThreshold is determined which call site should be an inlining candidate in AuxQueus. If the call site exceeds growth threshold, its priority list will be recalculated. i.e. the priority order list will be updated in the AuxQueue. As stated in col.7 lines 38-42 "If the priority of the best call site j is less than or equal to the calculated priority at block 412, then the checking is performed to determine whether the alternate best call site j from the AuxQueue should be inlined instead." thereby making the order dynamic.

2. In regard to independent claims 1, 8, 15 and 16, the cited references do not teach "determining the call sites to be inlined involves proceeding only once through the call sites in dynamically-updated globally-sorted working-list based order".

Examiner Responses:

The claim limitation "Proceeding only once" is supported in the original application by flow chart of Fig.5 and described on page 28 lines 24-28 "Thereafter, the determination is made 514 as to whether there are any more call sites on the working list to be analyzed. If so, then the process loops back to selecting 502 the most profitable call site remaining, followed by the legality analysis 504, and so on. If not, then the inline analysis phase 304 terminates and the inline transformation phase 306 is entered."

First, in Ayers' disclosure, it teaches "HLO performs several passes of inlining and cloning". It means using recursive procedure to select best call site for inlining. The algorithm is sketched in Figure 2.

On the other hand, Schmidt discloses two phases including approximation phase and refinement phase. Examiner only take refinement stage into considerations because Schmidt discloses only in the refinement stage that the determination of which procedures to inline possibly is changed and two priority queues are used to accomplish this (see col.6 lines 1-5). Also, as stated in col.6 lines 21- 25 "All leaf procedures are placed in the ready queue as indicated at a block 310. Then the best procedure is repeatedly selected from the ready queue and processed as indicated at blocks 312 and 314. First the best procedure is appended to the compilation order list as indicated at a block 316." From the foregoing description, Schmidt discloses determining the call sites to be inlined involves proceeding only once through the call sites in working list.

In addition, the claims are open-ended with the use of the term "comprising." The claims allow for proceeding only once during the approximation phase to generate the list and the proceeding once through said list during the refinement phase. The approximation phase is separate from the refinement stage.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Junchun Wu, Patent Examiner

Conferees:

Wei Zhen, Supervisory Patent Examiner

/Wei Y Zhen/

Supervisory Patent Examiner, Art Unit 2191

/Lewis A. Bullock, Jr./
Supervisory Patent Examiner, Art Unit 2193